First Lecture

September 4, 2023

Introduction:

Information theory deals with compression, transmission, storage and defence of information.
It gives answers to two quetions:
- what is the ultimate data compression: entropy
- what is the ultimate rate of transmission: channel capacity

Information theory is a branch of science that has a surprisingly clear starting point: it is the publication of a two-part article by Claude Elwood Shannon in 1948 entitled "A Mathematical Theory of Communication". Many basic results are already contained in this article.

At that time it was thought that increasing the transmission rate of information through a communication channel will increase error probability.
Shannon surprised:
- proved that it is not true if the communication rate remains below the channel capacity
- argued that random processes (e.g. music, speech) have a complexity below which it cannot be compressed. This is the entropy.
- said that while the entropy of the source is not greater than the channel capacity, error free communication is possible.

Naturally, many other important results were obtained during the more than 70 years that passed since then, and information theory today is considered a flourishing field of both mathematics and electrical engineering.

Two main problems in information theory are:
- Source coding: lossless (data compression) or lossy (when some distortion is allowed, e.g quantization)
- Channel coding

Goal of source coding: compressing data, that is encoding data with reduced redundancy.
Goal of channel coding: safe data transmission, that is encoding messages so that one can still correctly decode them after transmission in spite of channel noise. (This is achieved by increasing redundancy in some clever way.)

Short Probability Theory Revision:

random experiment
random variable
probability distribution
expected value

Notation: $\mathbb{P}(.)$ means probability, $\mathbb{E}(.)$ means expected value

Entropy:

Def. The *entropy* $H(P)$ of the probability distribution $P = (p_1, \dots, p_r)$ is defined as

$$H(P) = -\sum_{i=1}^{r} p_i \log p_i.$$

Immediate consequence of the definition: $H(P) \geq 0$, since $0 \leq p_i \leq 1$ and thus $\log p_i \leq 0$

Convention: $0 \log 0 = 0$

For $r = 2$ we speak about the *binary entropy function* of the distribution $P = (p, 1-p)$ and denote it by $h(p)$. Thus $h(p) = -p \log p - (1-p) \log(1-p)$.

Properties of $h(p)$: $h(0) = h(1) = 0$, $0 \leq h(p) \leq 1 = h(1/2)$

Convention: When no basis for a logarithm is given, we mean it to be of base 2.

The quantity

$$H_s(P) := \sum_{i=1}^{m} p_i \log_s \frac{1}{p_i} = \frac{1}{\log s} H(P)$$

is sometimes called the s-ary or base s entropy of P. (Note however, that the binary entropy function mentioned above is not the $s = 2$ case of this. Rather the $s = 2$ case of the s-ary entropy we simply call entropy in accordance with the convention that logarithms are to the base 2 if not said otherwise. If we really want to emphasize that $s = 2$ we might say base-2 entropy.)

Def. The *entropy* $H(X)$ of a random variable is the entropy of its probability distribution $P = (p_1, \dots, p_r)$, where $p_i = \mathbb{P}(X = x^{(i)})$ and the possible values of X are $x^{(1)}, x^{(2)}, \dots, x^{(r)}$.

Examples:

- Let X be a random variable with possible values: $\{1, 2, 3, 4\}$ and probability distribution $\mathbb{P}(X = 1) = \frac{1}{2}$, $\mathbb{P}(X = 2) = \frac{1}{4}$, $\mathbb{P}(X = 3) = \frac{1}{8}$, $\mathbb{P}(X = 4) = \frac{1}{8}$, then
$H(X) = H\left(\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{8}\right) = -\frac{1}{2}\log\frac{1}{2} - \frac{1}{4}\log\frac{1}{4} - \frac{1}{8}\log\frac{1}{8} - \frac{1}{8}\log\frac{1}{8} = 1.75 bits$

- Let Y be a random variable with possible values: $\{5, 6, 7, 8\}$ and probability distribution $\mathbb{P}(Y = 8) = \frac{1}{2}$, $\mathbb{P}(Y = 7) = \frac{1}{4}$, $\mathbb{P}(Y = 6) = \frac{1}{8}$, $\mathbb{P}(Y = 5) = \frac{1}{8}$, then
$H(Y) = H\left(\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{8}\right) = -\frac{1}{2}\log\frac{1}{2} - \frac{1}{4}\log\frac{1}{4} - \frac{1}{8}\log\frac{1}{8} - \frac{1}{8}\log\frac{1}{8} = 1.75 bits$, same as above, i.e. the values of the random variable don't play a role in the entropy.

- Let X be a random variable with possible values: $\{1, 2, 3, 4\}$ and probability distribution $\mathbb{P}(X = 1) = \frac{1}{4}$, $\mathbb{P}(X = 2) = \frac{1}{4}$, $\mathbb{P}(X = 3) = \frac{1}{4}$, $\mathbb{P}(X = 4) = \frac{1}{4}$, then
$H(X) = H\left(\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}\right) = -\frac{1}{4}\log\frac{1}{4} - \frac{1}{4}\log\frac{1}{4} - \frac{1}{4}\log\frac{1}{4} - \frac{1}{4}\log\frac{1}{4} = 2 bits$

Intuitive notion:
- "measure of uncertainty": how surprised we are when we get to know the value of a random variable
  how much information do we need to get rid of the uncertainty
- "measure of information": how many *yes-or-no* questions are needed to find out the message (assuming that we have the most efficient questioning system)

Some motivating problems:

1. What contains more information: the date of someone's birthday or the month of someone's birth?

Answer: Obviously the first one as it contains the second.

2. What contains more information: the month of someone's birthday or the name of the day of the week when this person was born?

Answer: The first one because it specifies one out of 12 possibilities, while the other specifies one out of only 7 possibilities.

3. How many yes-no questions are needed (in the worst case, but for the best strategy) to find a date of the year (someone's birthday, say)?

Answer: It is $\lceil \log_2 365 \rceil = 9$.

That many may be needed because each question partitions the set of possibilities into two subsets and if the answer always leaves us with the larger partition class then we will reduce the number of possibilities to 1 only after 9 questions.

That many is enough, because if we always ask questions that partition the set of still possible numbers into two subsets whose respective sizes are as close to each other as possible (i.e., they differ by at most 1), then after 9 answers we will have only one option remaining.

This argument shows that with 9 questions we can find one out of at most 512 numbers with similar rules. In general, if we have n options, then $\lceil \log_2 n \rceil$ questions are needed.

4. How many questions do we need if we must put all the questions at the same time, without knowing the answer to previous ones?

Answer: Somewhat surprisingly, the answer is the same. The situation is more restricted now, so we need at least as many questions as before. The point is that we do not need more. Here is an optimal strategy: write all the numbers (or dates, up to 365 or 512 or n, in general) into binary form and let the ith question be whether the ith binary digit is 0 (or 1). Clearly, this gives $\lceil \log_2 n \rceil$ questions and knowing the answer to all of them identifies the number (the date) we are looking for.

We will measure information content by the number of binary digits ("bits") needed to "describe" the information. Thus the information contained in telling one out of 512 numbers is 9 bits.

We feel that the above is plausible only if the probability of the possibilities is close to each other. E.g., learning the fact that one did not hit the jackpot in the lottery game this week seems much less information than learning that the same person actually did hit the jackpot. So we will have to take probabilities also into account.

Intuitive notion (again):
- "measure of information": how many *yes-or-no* questions are needed to find out the message (assuming that we have the most efficient questioning system)
We can write down the yes and no answers with $0-1$ sequences. These sequences uniquely determine the messages, i.e. they encode the messages.

Variable length source coding

Notation: For a finite set V, the set of all finite length sequences of elements of V will be denoted by V^*.

Model: Source emits sequence of random symbols that are elements of the *source alphabet* $\mathcal{X} = \{x^{(1)}, \dots, x^{(r)}\}$.

Given *code alphabet* $\mathcal{Y} = \{y_1, \dots, y_s\}$ (with s elements) we seek for an encoding function $f : \mathcal{X} \to \mathcal{Y}^*$ which encodes the source efficiently.

meaning of "efficient": it uses as short sequences of y_i's as possible, while the original $x^{(j)}$ will always be possible to be reproduced correctly.

meaning of "short": The average length of codewords should be small. The average is calculated according to the probability distribution characterizing the source: We assume that the emitted symbol is a random variable X and in the ideal situation we know the distribution of X that governs the behavior of the source.

Def. A *uniquely decodable* (UD) code is a function $f : \mathcal{X} \to \mathcal{Y}^*$ satisfying that $\forall \boldsymbol{u}, \boldsymbol{v} \in \mathcal{X}^*, \boldsymbol{u} = u_1u_2\dots u_k, \boldsymbol{v} = v_1v_2\dots v_m, \boldsymbol{u} \neq \boldsymbol{v}$ implies $f(u_1)f(u_2)\dots f(u_k) \neq f(v_1)f(v_2)\dots f(v_m)$ (where $f(a)f(b)$ means the sequence obtained by concatenating the sequences $f(a)$ and $f(b)$).

non-singular code: different source symbols have different codewords. It is not enough for uniquely decodability.

Prefix code: No codeword $f(x^{(i)})$ is a prefix of another. A prefix code is always UD.

Examples: (Codes given here with collection of codewords.) $C_1 = (0, 10, 110, 111)$ is UD, even prefix. $C_2 = (0, 10, 100, 101)$ is non-singular, but not prefix, not even UD, 100 can be $f(x^{(2)})f(x^{(1)})$ as well as $f(x^{(3)})$). But $C_3 = (0, 01)$ is UD, although not prefix.